2021 ASEE ANNUAL CONFERENCE
Virtual Meeting | July 26–29, 2021 | Pacific Daylight Time
Paper ID #34368

From UML Design to Implementation of a Reliable Student Information System

Briana Marie Bailey
Dr. Yujian Fu P.E., Alabama A&M University

Dr. Yujian Fu is an associate professor of computer science department at Alabama A&M University. Her research interests fall in formal verification of cyber physical systems, behavioral analysis of mobile security, software architecture and design analysis of safety-critical and mission-critical systems. Her projects are supported by NSF, Air Force and DoD. She have several publications regarding to the research and educational projects.

From UML Design to Implementation of A Reliable Student Information System

Author1 Name and Author2 Name
Author1 Affiliation/Author2 Affiliation

Abstract

When applying to undergraduate and graduate studies, adjusting to the new formats can be quite difficult. The Student Information System (SIS) is an intelligent and user-friendly Java-based system that allows the student and the professors to interact with various components within the system. The reliability of SIS is critical for the school system in order to maintain the correct information about academic records. Therefore, our project was established on the object-oriented software development cycle and UML based design concepts. With the object-oriented software development cycle of the requirement analysis, specification, design, implementation, and testing, we applied the UML design model on the programming of a middle size student information system. The use of an object-oriented approach based on UML specification can offer an understandable system representation, reduce the system's complexity, and improve the reliability and maintainability. We show the developed Java program is associated with the designed static and dynamic diagrams in the UML specification with the expected reliabilities.

Keywords

Object-oriented design, software development, reliability, student information system

1. Introduction

Many aspects in life have adjusted into a new technology-based world. As aspects become more digital in our every day lives, it was evident that our Universities/College were going to adjust as well. In transition to transforming into the new common society, campuses have organized and built several components that allow the students, professors, and faculty to access material through a designated internet browser. As a student who must abide by course room procedures such as deadlines, an interrupted server can cause a decline in the student's overall grade point. In response to unexpected difficulties, the System Information System (SIS), was configured to operate without direct internet source. In other terms, an effective, well-performed SIS can help bridge the gap between students and homework, students and professors, as well as professor to administration during an unexpected event. The SIS consist of similar components that can be found on a mainstream institutionalized website. Those components range from folders that contain various information that can be vital for both student and professor, registration for specific courses in order to retrieve the proper course material in a timely manner, as well as the ability to view content grades once it has been posted/submitted by the professor. While constructing the SIS, the overall objective was to construct a program that will continuously raise the academic success level, guide students through their career path, as well as being a beneficial

factor for the university/campus and those associated. Interaction among the associated members play a vital role in the overall operation of universities across the nation.

In UML model, a system can be described with various types of diagrams from different level and aspects of views. This leveled structures document the system in a set of systematic predefined diagrams – class diagrams, state diagrams, use case diagrams, sequence diagrams, activity diagrams, and so on. *Use Case Diagrams* describes the sequence functions' interactions with the externally visible behavior of the system – actors, e.g. users or different systems. *Class Diagrams* represent the structural aspects of the system with the expressing elements or entities of the respective of the software system. Several diagrams serves for describing behavioral aspects, including *State Diagrams*, *Use Case Diagrams*, *Activity Diagrams*, *Sequence Diagrams*.

In this class project, a SIS is designed in UML and developed using Java script, a in-browser safe programming language. Javascript can manipulate web page, web server, and interact with user. UML (Unified Modeling Language) is a de facto object oriented design language.

In the educational aspects, this term project bridges the gap between the high level OO design and software development concepts and the programming implementation. on top of it, it helps our learners to understand the design model, UML concepts, abstraction and application to the software development process. Finally, through this term project, our students will well understand how the design model guide the application as well as ensure quality software systems.

In the research aspects, through this project, an informal analysis of the design and implementation was performed. The validation method is performed manually against a set of reliable properties.

This paper is organized as follows. Section 2 introduces related works. Design models of the SIS are presented in Section 3. Followed by interface implementation of the SIS in Section 4 and System Implementation in Section 5. Discussion and conclusion are given in Section 6.

2. Related Works

For many institutes, a student self-server or student information system or SIS, typically encounters bugs and glitches due to poor configuration as well as an increasing number of students. With a vase amount of data, a student information system can experience system outages which can result in the student(s) not being able to retrieve specific information in a timely manner.

As SIS begins to increase throughout numerous campuses, an ongoing issue steadily arises. The cost to successfully build, test, and implement is relatively high. For most institutes, the cost can range from 500,000 to several million dollars. However, the main question is "Is the Student Information System worth the general cost?" The answer is yes. Although the student information system will temporarily affect the campus's budget, the enrollment and success rate of the campus will increase.

Our goal is not only to provide a smooth transition, but to also play a vital role in students' academic lives in a user-friendly manner. To create the SIS, various tools and panels were implemented to create the overall design and look of the system. Although various student information systems have positive results, it's important to recognize common fault/flaws within the innumerable systems. Many researchers focus on the positive affects within the system, however students within research [5] have analyzed and evaluated the various flaws. One evident flaw among SISs are their constant need for maintenance. Many programs exceed 1,000 lines of code, which means the need for regular maintenance repairs are relatively higher. Without the proper evaluation, the system will experience interruptions more often than intended. Along with the high need of maintenance, studies also show that many students have little to no knowledge on how to properly operate the newly implemented systems. [5] elaborately described how numerous operational workshops have done little justice in reference to resolving the issue. In the Student Information System intended for AAMU, devoted time has been in place in means to ensure regularly scheduled maintenance, user-friendly instructions in terms of daily operation, as well as ensuring an increase of academic scores among student that use the system. The below figures below represent the several interfaces within the SIS as well as the various components that the students, faculty, and staff will have access to.

Throughout the decade, there have been numerous researches conducted regarding student information systems. As the generalized named may range greatly, the concept and goal remain the same. Universities across the global has implemented or are currently implemented systems to increase their student academics rates. In terms of past researches, in 2011, a group of intelligent students with Olayan School of Business, constructed a web-based decision support tool for academic advising in hopes to solve an ongoing issue within their universities technology-based advising system [1]. In comparison to the Student Information System, their academic advising system was implemented to increasingly help the student prosper throughout their undergraduate and/or graduate years. In their research, [3] concluded that, "Research indicates that usability of a web site is associated with numerous positive outcomes, including reduction in the number of errors, enhanced accuracy, and more positive attitude towards the target system, and increased usage [2].

In recent studies, researchers clarify the overall purposes and need for an academic advising system. As many may know, academic advising is becoming relatively time consuming for both the students and professors. As the number continuously increases in the enrollment status, its vital to ensure that the appropriate system are in place. In reference to the researchers with the University of Tabak, there are practical cases that are relatively too large to continuous change the required timetables [3].

In many studies, it is vital to view both the positive and negative features within constructing a Student Information System. Within their research, [4] locates and elaborates of the negative impacts that the system can have on both the student and staff as well as detailing their methodology as to how the information was generated. In order to understand the faults/cons, the content will be placed within a pro then con format. As stated previously, implementing a student information system results in a higher effectiveness, increase in productivity, as well as ensuring safety and overall satisfaction [4]. However, as stated previously, with every pro there

comes a possible con. According to [4], one vital con with student information systems are the maintenance requirements. Due to the system having a specified deadline, most of the maintenance in the beginning process is rushed, which result in future work needing to be done. A typical system can have over 1,000 lines of code; therefore, every line will eventually need to be repaired and may be poorly documented [4]. Resulting in maintenance being one leading con and occasional pro of student information systems. Student Information System's are projected to allow students to take a leap in their education as well as along them to be aware of what is going on within their academic career. However, the lack of end-user training was identified as one con towards SISs [4]. Although the SIS was constructed to be user-friendly as well as having numerous usability feature. Based on their campus results, students were having numerous issues understanding how to properly operate the SIS even after a verbal orientation [4]. As of [5], who also conducted a research pertaining to the faults of SIS, many cannot believe how many companies show a lack of interest in the generalized value of appropriate training for students.

3. Design Model of The SIS

A clear and descriptive design model was configured to illustrates the features within the Student information System. Static and structural aspects are represented in classes and class diagrams, state transitions of critical classes are described in state diagrams. The system high level representation is shown in use case diagram, while the object interactions are described in sequence diagram. Workflow and algorithms are described in activity diagrams. Functionally, the SIS consist of approximately six interfaces. Each of those interfaces contain various functions that allows the system to operate in the intended manner. Please label class diagrams, state diagrams, use case diagrams (missing), and sequence diagrams.

3.2. Use Case Diagram

Use case diagram shows the high level abstraction of sequence of functions of the system interactions with external entities. It helps the developers to analyze the overall system functions and communications. Our use case diagram (Fig. 1) shows two layers of use cases related with include and extends relationships. Three actors – Students, Professor, and Admin – handle two layers of use cases. Layer one use cases interacts with actors student and professor including account, course registration, grading and advising. Layer two use cases interacts with those in layer one and actor admin. These use cases handles the functions of backend course posted, update and withdraw. The design of use case diagram abstracts the high level representation of system functionalities interacting with external actors. This use case diagram helps students to understand the subsystem design and organization. From use case, it will be able to elaborate individual classes and derive other dynamic diagrams.

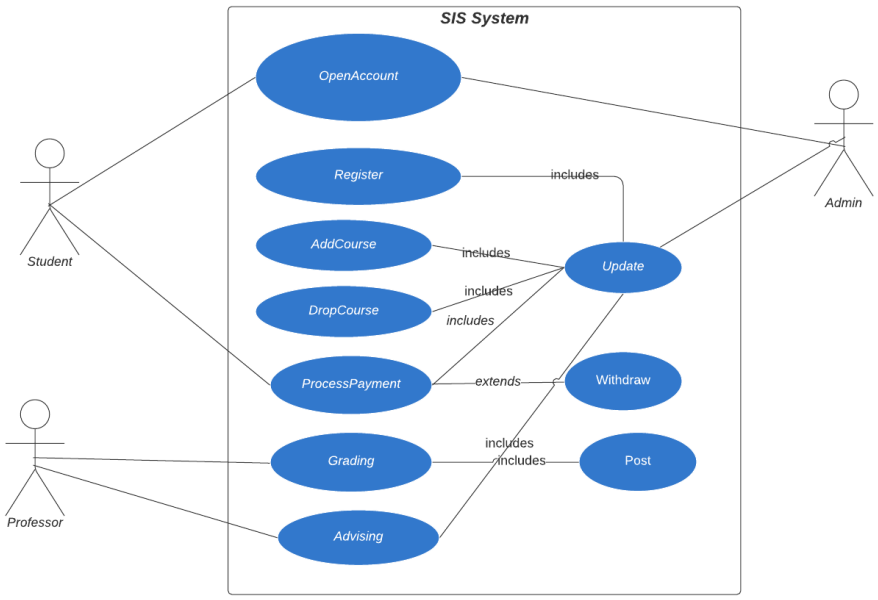
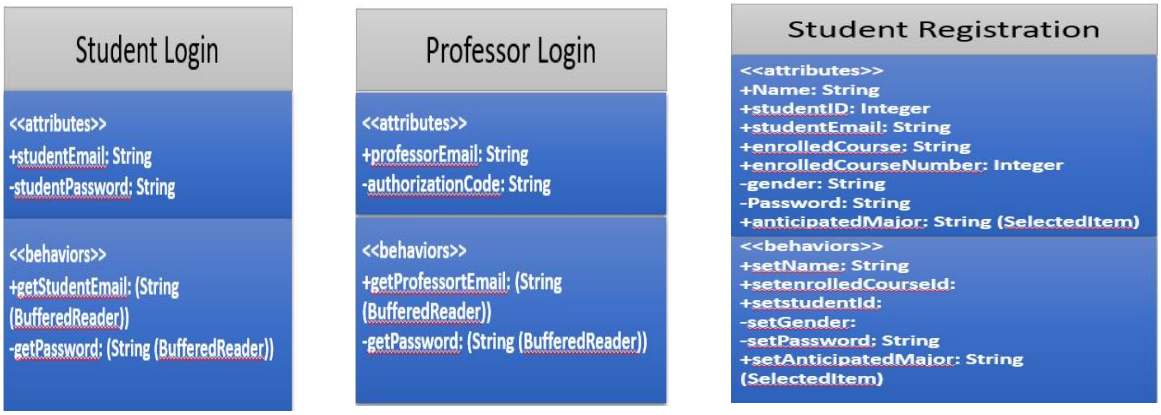


Figure 1. Use Case Diagram of SIS System

3.1. Class Diagram

Class diagram shows the structural organization of the system entities and their relationships. In the SIS, we considered eight (10) classes, Person, Student, Professor, Admin, StudentLogIn, ProfessoryLogin, Course, Grading, StudentRegistration, Payment and the aggregate class StudentInfSys. While classes Student holds the information of each individual student profile



(a) (b) (c)

Figure 2. Some classes defined in UML (a) Class Student Login; (b) Class Professor Login; (c) Class Student Registration.

which includes person info, academic records, financial info, etc.. Class Professor carries the professor's individual info, teaching assignment, and grading. As this is a small version that focuses only on the academic record of student performance at a mild level, some information such as faculty and stuffs salary are not considered in this design. Similarly, class Admin carries the individual info of stuff but focuses only the functions of student records and registration handling. All the three classes (Professor, Student, and Admin) are generalized in a super class Person.

Class Course holds the course title, course number, credit hours, location, schedule and instructors, etc.. Grading class is designed to handle the student assessment regarding a class where a professor who is assigned as instructor of it can access. This class takes the reference from both object Professor and object Course. It is noted that the grading is a simplified version where some aspects of the class performance are not included. Class Registration handles the student registration, issue the payment, and send the confirmation. Class Payment will take the student financial information and course information based on the student's registration to calculate and process the payment for each semester. All these three classes are designed to associate classes for the purpose of isolating the data flow from the references, increasing the data encapsulation and reducing the system complexity. Some selected user-defined classes are shown in Figure 2. The overall class diagram is shown in Figure 3.

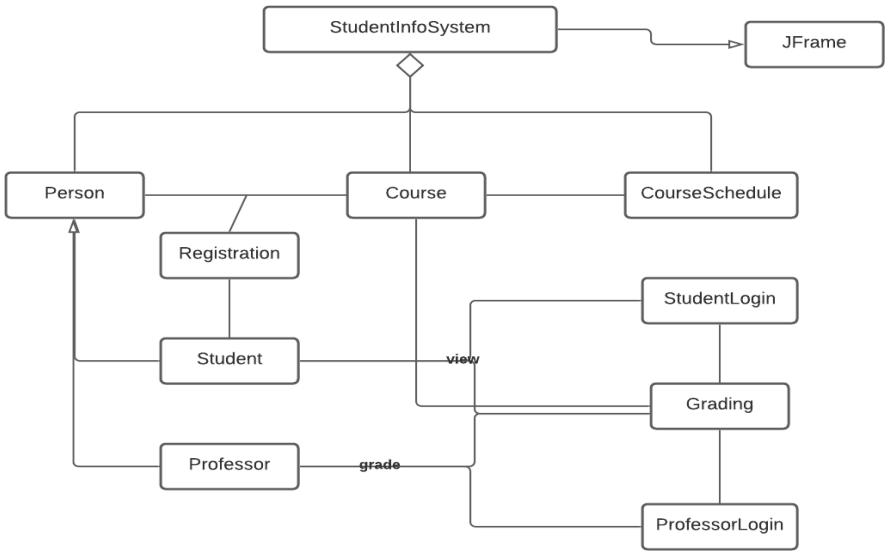


Figure 3. Class diagram of SIS.

The overall class diagram considered the user defined class and some of the pre-defined classes from the library. From the class diagram, it is easy to tell the registration functions based on the information from object of student and courses. The payment process is inherently complex in which we simplified by considering the financial data from student object and courses only.

3.3. State Diagram

State diagram shows the object state transitions with the receiving signals. In our designed system, each class has an associated state diagrams to document the behaviors. In this section, two state diagrams will be presented: (1) Registration state diagram; (2) Professor state diagram. Each of them will be described below.

The diagram in Figure 4 demonstrates the transition between student registration where student login and the student's information are considered. Each interface page requires information from the student based of the specific action being pursued. Student registration's main function pertains to inputting information into a secured file that solely contains the student's provided information. This feature uses the primary function file writer and buffered writer to retrieve the information from the interfaces text fields. Within the student registration form, the student must submit their email as well as a password. With those two components, the student is able to access the student login portal. This portal uses the functions of file reader and buffered reader to cross-match the inputted data with the stored data. If the input matches the stored data, the student is able to proceed towards the student's home page. However, if the data does not correlate, an error message will appear encouraging the student resubmit information. With the resubmission, the page will take the student back to the registration page. Once the information has correctly been submitted and approved, the student's home page will appear which leads them to a separate screen. Within the home page section, the student is able to view homework that has been assigned, submit homework, as well as searching content related to the course and their professor.

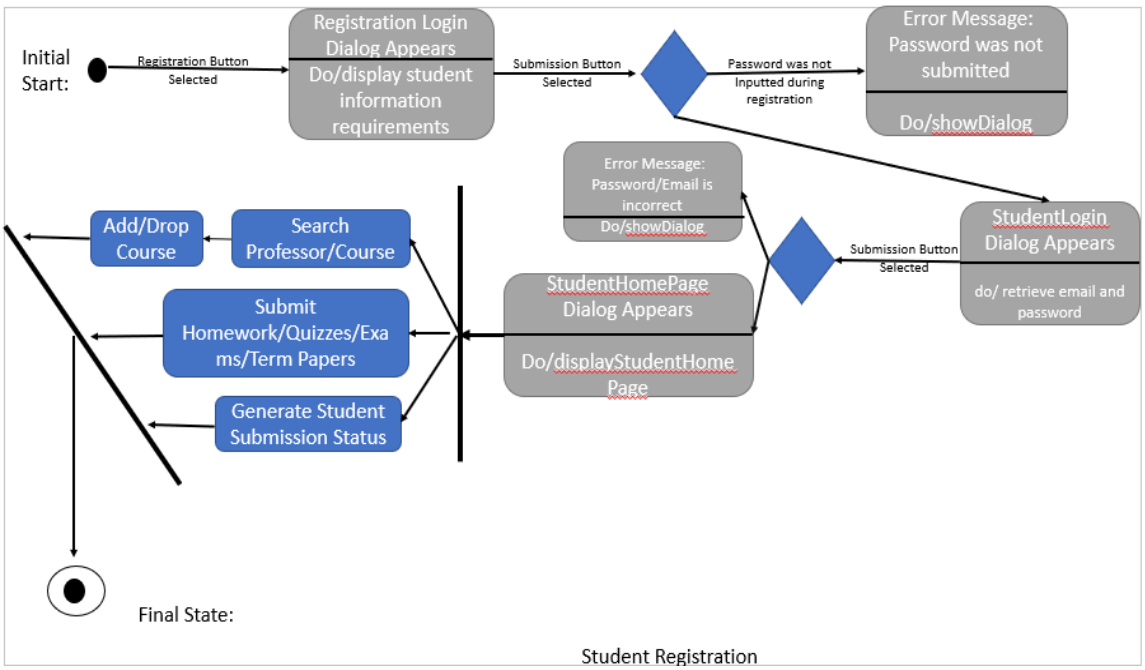


Figure 4. The state diagram of Registration.

The Student Information System consists of various components and transitions incorporated to ensure efficient operation. Figure 3 illustrates the generalized transitional path through the professor's course log. Any content pertaining to students, professors, and administrators must be

safeguarded to ensure only authorized personnel have access. Upon reaching the course log, the professor must log in with their provided credentials. However, if the credentials submitted are incorrect, an error message displays to the end-user declaring that the information is wrong as well as suggesting that the end-user attempts their credentials again. Once the information is verified, the professor will then have the ability to view the content within the course log. The course log allows the professor to assign content (such as homework, quizzes, term papers, discussion board, etc.), upload student grades, create and modify deadlines, submit and modify course syllabus, as well as checking the submission status from each student.

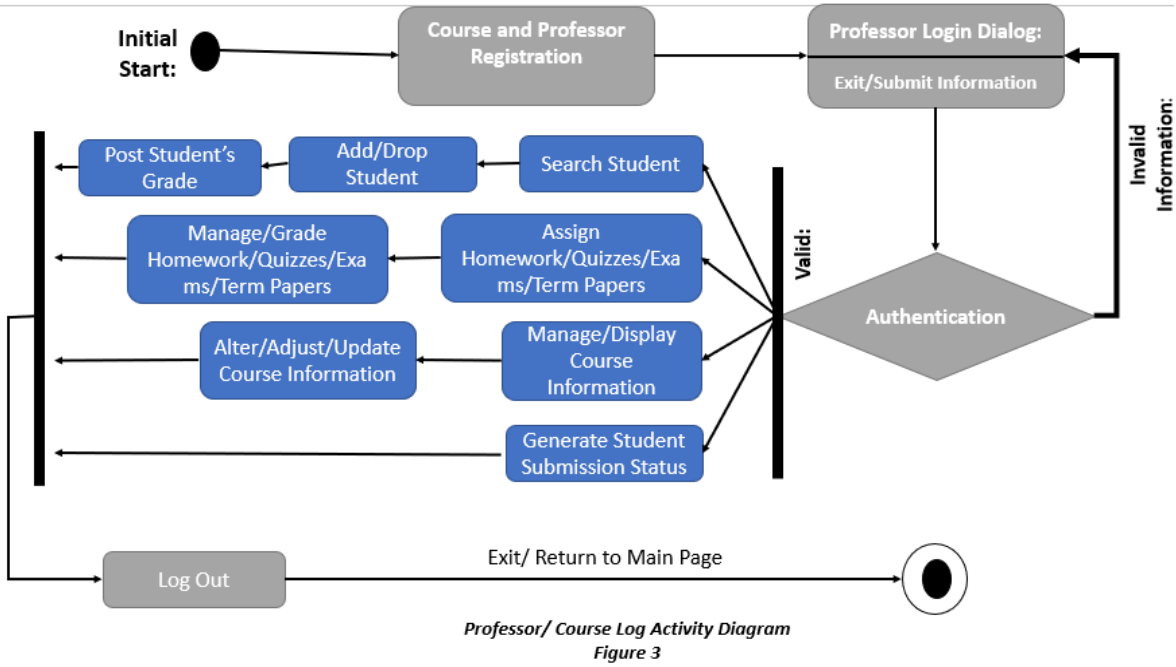


Figure 5. The state diagram of Professor.

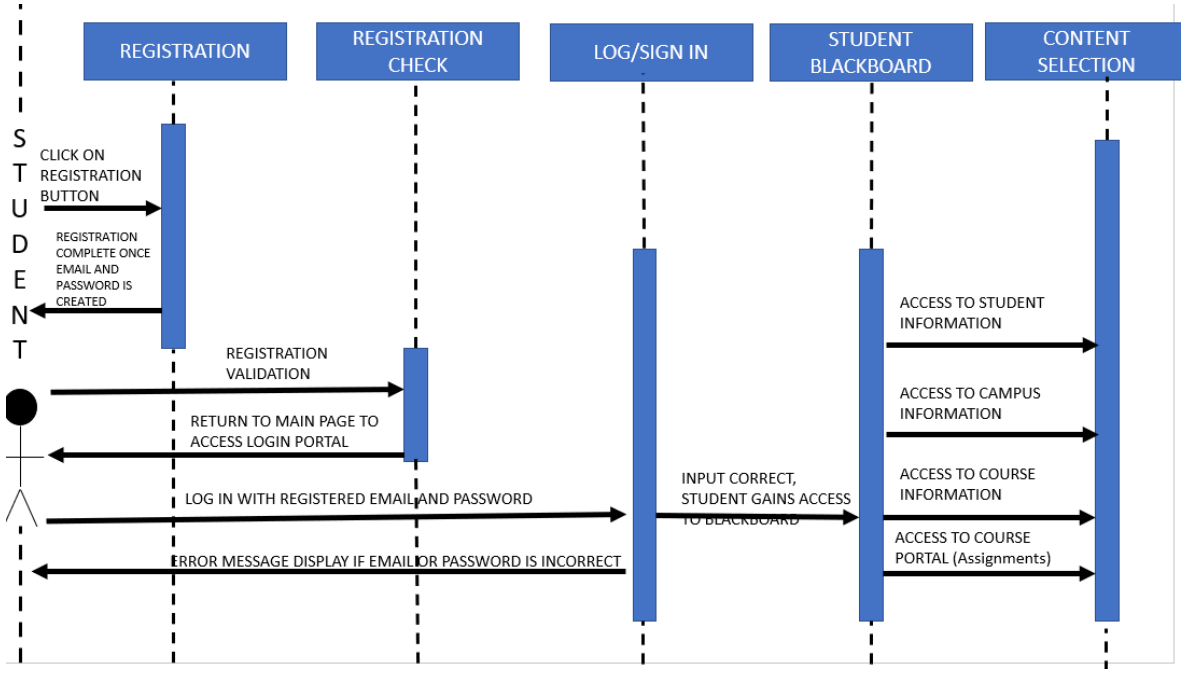


Figure 6. The sequence diagram of SIS.

3.4. Sequence Diagram

As stated previously, the SIS consist of transitions, conditions, components, and task accomplishable described by the above UML Models. To go in depth of their components and methodology, the following provided images elaborate on the abundant classes. Sequence diagram can show large scale system interactions regarding different scenarios. Cross control with the student data and view from professor, student, and admins, the sequence diagram is able to demonstrate each session with the different events. Each use case requires one or more sequence diagrams to describe its behaviors. Multiple use cases can be represented in a very large sophisticated sequence diagram with structured messages. For our mild SIS, Figure 6 is a representation of a sequence diagram regarding the student and accessibility.

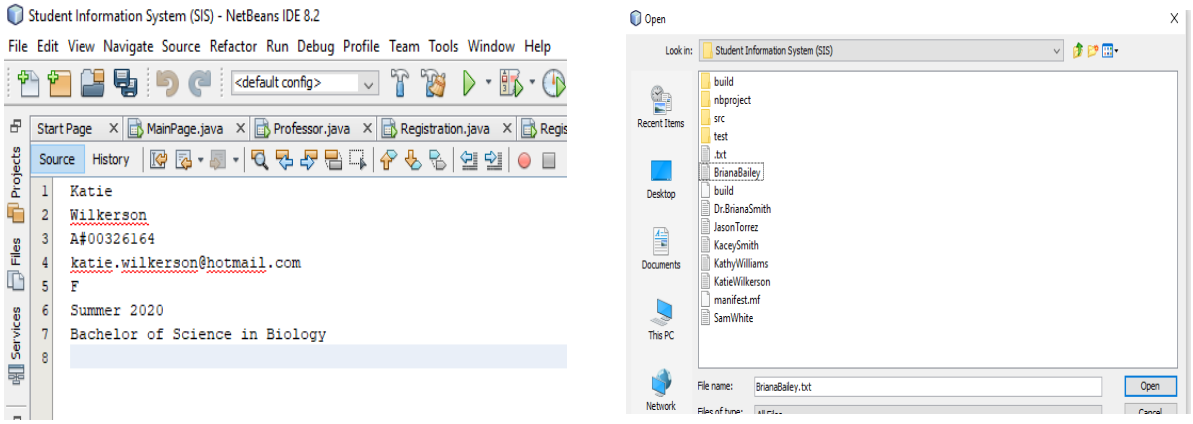


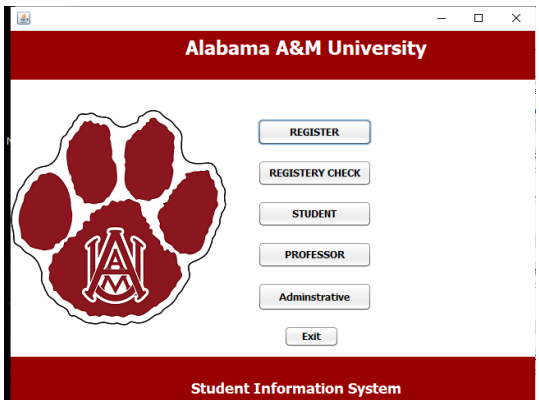
Figure 7. The content of student information in Java palette in NetBeans.

4. The Interface Implementation of SIS

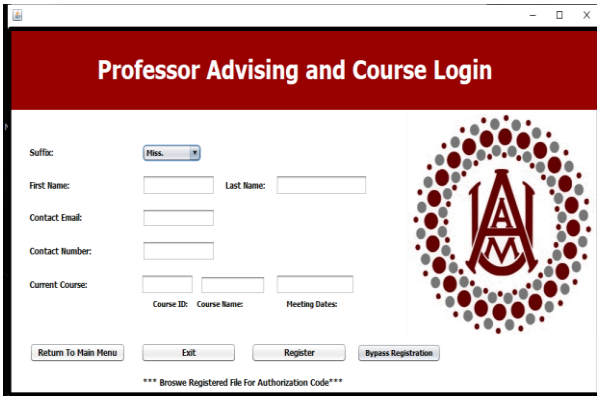
While constructing the Student Information System, various tools and applications were employed to achieve the desired outcome, appearance, and performance. Due to the SIS requiring different interfaces, the Java Scripted Program known as NetBeans was used to achieve maximum quality and efficiency. NetBeans allows the end-user to construct interfaces with a wide variety of tools within a given palette, such as tabbed panels, menu bar, etc. The palette allows the end-user to have a variety of options to incorporate into their overall design. The tool was used throughout every interface; however, the most evident use is within the initial greeting interface. The greeting interface is simply the initial, eye catching, interaction panel. Within this panel, a student or professor has the option to navigate between different interfaces. For a student to access their information, they must first begin by registering their information into the system. By registering, the student will input basic information, such as their first name, last name, student identification number, student email, etc. Once the information inserted into the tooled text box, it will then save within their own personal and private folder. The folder can only be accessed by the student and the student's current course assigned professor. In order to properly store the student's information, a file writer and buffer writer syntax was in embedded into the source code within a provided tool button renamed as "register". To elaborate more efficiently, an example syntax follows:

FileWriter writer = new FileWriter (firstName.getText() + lastName.getText() + ".txt");
BufferedWriter br = new BufferedWriter(writer);
Writer.write (firstName.getText() + ".txt");

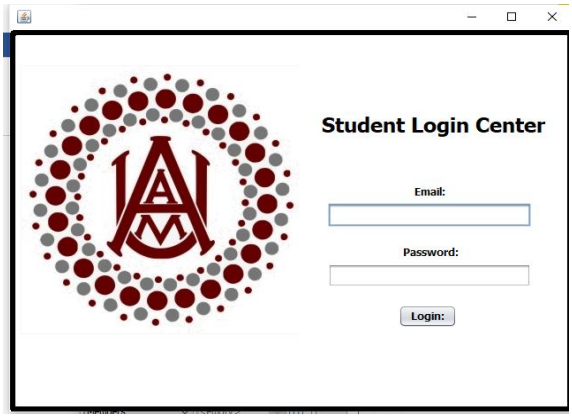
Those provided statements allows the system to retrieve the data input within the various text box tools provided within the interface. Each input has a designated code variable that will allow the information to stay organized and maintainable once it is secured within the folder/file. As stated above, the previous information is all contained within one specific tool referred to as a button/toggle button. Although there are numerous buttons within the interface, the registration/save toggle button is the most beneficial due to the need of the student's information for further action.



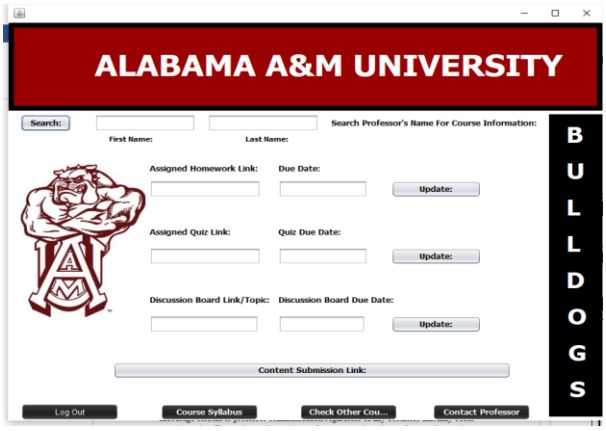
(a)



(b)



(c)



(d)

Figure 8. Some sample user interfaces (a) homepage; (b) Professor advising & login; (c) Student login page; (4) Course information search.

The NetBeans Java palette with the properties panel for coding purposes consists of the various components implemented into the Student Information System. The different components bring a new look onto the interface and design of the system. Once the specific components, such as text box, are selected, the properties tab allows the programmer to assign a variable name. Altering the variables name from its default name, allows the coding and overall build to relatively easier due to its exclusive name.

Once the student registered to the SIS, the student profile will stored with the necessary information, which is shown in the interface of Figure 7. The professor will be able to access the student's information for educational purposes. In terms of the student, for the learning purpose, once the student has registered and has searched for their professor, the student will be able to view the professors assigned content (homework, quizzes, discussion boards, etc.), contact information (email and phone number), as well as their course meeting date and office hours.

The overall system interface design integrates the information collection, performance and record, and advising. The combo box was implemented to allow the student to have an easier navigation between various semesters, as well as the available courses correlated with the selected semester. In regards to the panel, it can be classified to be more beneficial for the programmer/designer than towards the students and professors. The panel acts as an anchor for the various tools used, can be manipulated in appearance, as well as keeping all material within to reduce complexity and confusion for others. Unlike the panel, labels can be beneficial for both programmer and end-users. The labels act as a guide to demonstrate what goes where, how it supposed to be entered, as well a design essential. Sample interface design is shown in Figure 8.

```
class StudentRegistration {
        Student student;
        Course course;
        boolean registration_status;
        String holdType;
        String classify;

        public StudentRegistration(String studentFirstname, String studentLastName, int studentID,
String email, boolean gender, String password, String enrolledSemester, String majorSelection) {
        // initialize the object student registration with the parameters.
        }
        public void setStatus(boolean b) { registration_status = b;}
        public boolean registration() { student.registrar = 'R';}
        public void setHold(String h) {holdType = h;}
        public void setClassify(String c) {student.classify = c;}
        … …
}
```

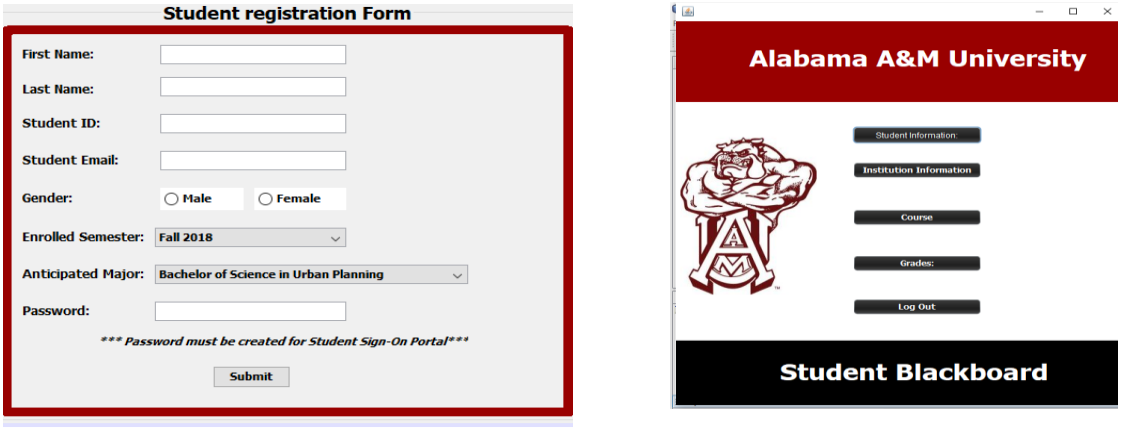
Figure 9. Java Implementation of class StudentRegistration.

5. System Implementation

Within the constructed SIS, student information system, there are numerous functional implementations incorporated to ensure proficient result. The system implementation section will elaborate on the various patterns and components inserted within each sub-section of the program. This will act as a pseudocode in means to reduce the code drastically. As stated previously, there are several classes employed such as, student, professor, student login, professor login, registers (both student and professor), student home page, and etc. Following each component's class description, each interface will be provided to represent a visual representation of the system. The common pattern within each class is the need to extend to a different class for various components, as well as a get and set syntax in order to retrieve, set, and display information from the extended class. Some classes are programmed and described below.

Within class StudentRegistration, the student is able to enroll onto the Student information System and create their own personal file. Within this file, all inputted information is stored and only

accessible by the student and the professor of the student's enrolled course. The interface of StudentRegistration is shown in Figure 10. Class StudentLogin is shown in Figure 11.



(a) (b)

Figure 10. Java Implementation of class Student Registration (a) and Student Homepage (b).

```
class StudentLogin {
                public String email;
                private String password;
                FileReader read = new FileReader(email.getText());
                BufferedReader content= new BufferedReader(read);
                public emailStudent = content.readLine();
                String passStudent= content.readLine();
                public boolean validateUser(String urn, String pw) {
                    if(emailStudent.equals(urn)&& passStudent.equals(pw){
                        // allows the student to login}
                    else {
                        System,out.println("Incorrect Email or Password! Please try again");
                 }
            }
}
```

Figure 11. Java Implementation of class Student Login.

The StudentLogin section allows the system to read the information inputted by the student and match it with the information provided by the student during registration. If the information is correct, the student will able to proceed to the student login, however if the information is incorrect an error message appears advising the student to attempt to login again. The interface of the student login class is shown in Figure 10 (b).

The StudentHomePage (Figure 10 (b)) is the following screen that the students can access once successfully logging in. The home page is where the student will be able to view the course

information, the assigned homework by specific teachers, as well as being able to view the assigned due date. This component was implemented in hopes to allow the student to remain on track throughout their academic year.

The StudentInformationSystem is the aggregate class where each key object will be instantiated as well as the pre-defined objects of swing package. In this section the students and faculty start to navigate through our provided content. Although the components aren't directly restricted, each individual must go through a login interface that requires them to input specific information for access.

To handle professor login, each professor is provided with an authorization code within their file that must be used in order to create their own password. The professor login and ongoing interfaces can only be accessed if the inputted email and password is correct. If the information does not match what is on file, the professor will receive an error message stating that either the email or password does not correlate with existing accounts.

The last section designed for the professor is their course log in. Within the professor course portal, they have the ability to assign content, grade content, as well as viewing the submission status from each student. It is vital for the student and professor to have a clear understanding regarding the course and the professor's academic expectation for each student. Therefore, both the professor and student have the ability to view the course's syllabus. Due to the professor course log having a large variety of code, the class will not be provided. However, the below content will display visual representation of the professor's ability.

6. Discussions and Conclusion

During this term project, we have applied several main UML diagrams in the support of Student Information System design and development. Reliability is one of the key desired properties of SIS. To ensure the system meets the desired reliability, we define any active or passive object will respond to the events (signal, change, or time events) eventually. To analyze the SIS hold this property, we look into each state diagram of each key objects (registration, payment, and grading) to observe the state transitions in response to the events correctly. In addition to that, for the UML model analysis, OCL (Object Constraint Language) can be defined with the class diagrams and states. With the defined OCL, the proper verification tool can be adopted to analyze the reliability systematically and automatically. Due to time limit, in this study, we only focus on the first approach.

The developed SIS based on the UML diagram has been tested for various course registration, class grading and advising. All operations of the registration, adding/removing courses, grading can be conducted correctly with the input values. Advising page shows the faculty advisor selection and allows the student to make an appointment. The SIS demonstrated an efficient response and supporting for student in academic activities.

In the past decades, UML has been widely adopted and applied to various application domains. It is very powerful tool and provides numerous modeling elements needed for the special purpose under examination. From the educational purpose, it is an efficient way for the project based learning in the master of conceptual and abstract content in STEM [6]. Through this term project,

we learned and well understood the key UML diagrams and how to build the UML model with the related semantics as well as avoid undesired complexity. Furthermore, a systems development process for object based information systems which builds on UML is to be defined to use UML both efficiently and effectively.

The student information system is projected to ensure the student's ability to self-service themselves as well as bettering their overall experience through their high school to college or college to graduate school transition.

References

1 Feghali, T., Zbib, I., & Hallal, S, "A Web-based Decision Support Tool for Academic Advising. Educational Technology & Society", 14 (1), 82–94, 2011.

2 Nielsen, J. (2001). Users first-web usability: why and how, retrieved December 10, 2010 from http://www.zdnet.com/devhead/stories/articles/0,4413,2137433,00.html.

3 Abedlhamid, A. Ayoub, Alhawiti, "Agent-based Interlligent Academic Advisor System", ISSN 2319-7900, April 2015.

4 Ariel G., M. Rebortera, "Issues and Concerns in the implementation of the Student's Information System", Knowledge E DOI 20.18502/kss.v316.2378, 2017.

5 Artit, K., "Management Information System Implementation Challenges, Success Key Issues, Effects and Consequences: A case Study of Fenix System.", Jokoping Internal Business School, May 2012.

6 Hall, A. and Miro, D. (2016), A Study of Student Engagement in Project-Based Learning Across Multiple Approaches to STEM Education Programs. School Science and Mathematics, 116: 310-319.